WALAILAK JOURNAL
Applied Sciences
http://wjst.wu.ac.th https://doi.org/10.48048/wjst.2021.13109

Deep Transfer Learning Pipelines with Apache Spark and Keras TensorFlow combined with Logistic Regression to Detect COVID-19 in Chest CT Images

Houssam BENBRAHIM[1,*], Hanaa HACHIMI[2] and Aouatif AMINE[1]

[1]*Engineering Sciences Laboratory, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco*
[2]*Systems Engineering Laboratory, Sultan Moulay Slimane University, Beni Mellal, Morocco*

([*]Corresponding author's e-mail: houssam.benbrahim@uit.ac.ma)

Received: 26 September 2020, Revised: 19 March 2021, Accepted: 26 March 2021

Abstract

The SARS-CoV-2 (COVID-19) has propagated rapidly around the world, and it became a global pandemic. It has generated a catastrophic effect on public health. Thus, it is crucial to discover positive cases as early as possible to treat touched patients fastly. Chest CT is one of the methods that play a significant role in diagnosing 2019-nCoV acute respiratory disease. The implementation of advanced deep learning techniques combined with radiological imaging can be helpful for the precise detection of the novel coronavirus. It can also be assistive to surmount the difficult situation of the lack of medical skills and specialized doctors in remote regions. This paper presented Deep Transfer Learning Pipelines with Apache Spark and KerasTensorFlow combined with the Logistic Regression algorithm for automatic COVID-19 detection in chest CT images, using Convolutional Neural Network (CNN) based models VGG16, VGG19, and Xception. Our model produced a classification accuracy of 85.64, 84.25, and 82.87 %, respectively, for VGG16, VGG19, and Xception.

Keywords: COVID-19, Deep Transfer Learning Pipelines, CNN, Apache Spark, Logistic Regresion

Introduction

The outbreak of the SARS-CoV-2 named COVID-19 [1] has been causing global concern. It came from Wuhan, China, in December 2019 [2] and has spread to almost 223 territories, including Morocco. On March 11, 2020, the World Health Organization (WHO) declared the novel coronavirus (2019-nCoV) as a global pandemic. Globally, on March 14, 2021, there had been 119,220,681 confirmed cases of COVID-19, including 2,642,826 deaths, reported to WHO [3]. In Morocco, several coronavirus cases have been marked. On March 14, 2021, the Moroccan health ministry [4] announced that the cumulative number of confirmed cases of SARS-CoV-2 was 488,937, of which deaths were 8,723, and the number of cured patients were 475,849.

The people infected by the 2019 novel coronavirus showed symptoms like fever, cough, dyspnea, headache, muscle soreness, and fatigue [5]. Several laboratories in the world employ Real-time Reverse-Transcription Polymerase Chain Reaction (RT-PCR) for detecting, tracking, and studying the COVID-19 in suspected cases [6]. However, recent research has demonstrated that RT-PCR has a sensitivity of as low as 60 - 71 % for detecting COVID-19 [7-9], which can probably be assigned to laboratory error [10]. Computed Tomography (CT) test of the chest is one of the techniques used to diagnose pneumonia. Multiple studies have been demonstrated that the use of deep CNN for the detection, quantification, and monitoring of COVID-19 in chest CT [11,12] or X-ray [13-15] images present important results. Indeed, the use of new technologies in medical imaging allows it to take a paradigm shift. Due to Deep CNN

based on new tools such as Keras TensorFlow, we can easily automate the detection and the classification of COVID-19 in chest CT images [16], which allows doctors to make a better diagnosis and to devote more time to patients and their needs, and above all to overcome the problem of the lack of medical skills and specialist doctors in remote areas.

The general goal of this article is to create an architecture of Deep Transfer Learning using CNN pre-trained models VGG16, VGG19, and Xception founded on advanced technologies like Apache Spark framework and Keras TensorFlow coupled with the logistic regression algorithm. This solution allowed the doctors to obtain a clear view of the presence or absence of COVID-19 in chest CT images with high accuracy and certainty, encouraging Moroccan medical competencies to use its advanced tools as a diagnostic technique in medical imaging.

Machine Learning Algorithme

Machine Learning (ML) is a technique of data analysis that automates the development of analytical models. It is a subfield of Artificial Intelligence (AI), which can reproduce behavior through algorithms fed by a large amount of data [17]. The idea is that the algorithm learns which decision to take and creates a model. This means that the machine can automate tasks depending on the situation [18]. ML is used for multiple domains such as recognizing objects, natural languages, diagnostic assistance, bioinformatics, fraud detection, cybersecurity, financial analysis, search engines, brain-machine interfaces, software engineering, robot locomotion, and predictive analysis in legal and judicial matters [19,20]. The learning algorithms can be categorized according to the learning model they use, such as supervised, unsupervised, semi-supervised, partially supervised learning, reinforcement, and transfer learning. Logistic regression (LR) is a popular supervised classification algorithm in ML [22]. LR is a statistical approach used to evaluate and characterize the relationships between a binary type response variable, Y, and one or more explanatory variables, which can be categorical or continuous numeric type, X [23]. Logistic regression is used when the dependent variable has only two values, such as 0 and 1 or "Yes" and "No" \ cite [24]. LR can be combined with the Apache Spark framework to build a machine learning classifier model.

Apache Spark

Big data analytics is the concept of examining massive data sets containing heterogeneous data types to uncover hidden patterns, unknown correlations, and other actionable information using advanced analytic methods [25]. To perform the computational requirements of large data analysis, a powerful framework is fundamental to design, implement, and manage the required algorithms and techniques such as machine learning, deep learning [26]. For this reason, Apache Spark has appeared as a unified engine for large-scale data analysis. Apache Spark is an open-source de facto framework for big data analytics [27]. It is differentiated by its speed, ease of use, and sophisticated analytics. Spark runs in memory, on clusters. It is the next-generation engine for big data analytics after Hadoop's MapReduce. Apache Spark can run standalone, on YARN, Mesos, and in the cloud, where it can read data directly from Hadoop Distributed File System (HDFS), Hbase, Cassandra, Hive, and Tachyon [28]. It supports multiple languages Java, Scala, Python, and R, and it comes with several libraries Machine Learning (Mllib), SQL (Spark SQL), Graph, and Parallel Graph (GraphX), and Streaming (Spark streaming). Spark core is founded upon the Resilient Distributed Datasets (RDDs) abstraction to store data in memory (RAM) [29]. Apache Spark is a key framework for deep learning, and Keras TensorFlow can be attached to Spark workflows to create an advanced performance for image classification solutions.

Deep Convolutional Neural Network

Deep learning (DL), deep structured learning, or hierarchical learning is a part of ML methods based on Artificial Neural Networks, which is used to model data abstraction based on articulated architectures of different nonlinear transformations [30]. DL applies to various sectors in bioinformatics, recommendation systems, customer relationship management, drug discovery and toxicology, natural language processing, image recognition, and automatic speech recognition [31]. The Principe of DL is that the machine can learn without having to be programmed, in contrast to the principle of traditional programming in which the program executes rules, instructions, and operations [32]. DL uses algorithms

that can be supervised or unsupervised, and they include pattern recognition and statistical classification. DL is based on several layers of processing unit extraction and transformation of characteristics. DL works with learning on several levels through various layers; one passes from low-level parameters to higher-level parameters [33].

The pipeline or a data pipeline is a chain that connects a set of transformations that are arranged according to a certain order when we are dealing with data and making different operations on it. The main aim of a data pipeline is to permit us to assemble and manage the datasets required for model training. This means getting the data into a form that the model can support and comprehend [34]. An ML pipeline is used to automate ML workflows. They function by allowing a sequence of data to be changed and correlated together in a model that can be tested and evaluated to attain a positive or negative result. A pipeline consists of a sequence of steps. ML pipelines are repeated as every stage is iterative to continuously enhance the model's accuracy and reach a successful algorithm [35]. Deep Learning Pipelines is an open-source library founded by Databricks that furnish high-level APIs for scalable deep learning with Apache Spark in Python. DL Pipelines were created on Apache Spark's ML Pipelines for training and Spark DataFrames and SQL to display models [36].

In ML, a Convolutional Neural Network (CNN) called ConvNet is a category of Artificial Neural Networks, such as the connection motif between neurons, which is influenced by the visual cortex of animals [37]. It is a DL algorithm employed in text classification, speech recognition, document analysis, human pose estimation, action recognition, scene labeling, face recognition, and image classification [38]. With CNN, we can automatically detect the presence or absence of COVID-19 in CT images [11,12]. CNN consists of trainable multistage architecture with every phase composing of multiple layers. The input and the output of each phase are called feature maps. Each stage generally includes a Convolution layer, a Non-Linearity layer, and a Pooling or Sub Sampling layer. A single or several Fully Connected Layers (Classification) are present after multiple Convolution and Pooling layers [39].

From the late 1990s up to 2020, multiple amelioration in CNN learning architecture and methodology was implemented to make CNN scalable to large, complex, multiclass, heterogeneous problems. There are different types of CNNs architectures such as Xception, GoogleNet, Alexnet, ResNet, VGGNet, and InceptionV3 [40]. In this work, we used 3 CNN pre-trained models to detect COVID-19 in the CT images, which are:

- **VGG:** VGG is a CNN architecture that was used in the ImageNet Large Scale Visual Recognition Challenge ILSVRC competition in 2014. This model was proposed by Simonyan et al. from the University of Oxford in 2014 in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition" [41]. The pre-trained networks VGG16 and VGG19 are "16" and "19" weight layers deep in the network, respectively. VGG attains 92.7 % top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1,000 object categories, such as a pencil, mouse, keyboard, and many animals. VGG replaced the 11×11, 7×7, and 5×5 filters with very small 3×3 filters over the whole net, which is convolved with the input at every pixel. For instance, the experimental results demonstrated that a stack of 3×3 filters could induce the effect of the large size filter (5×5 and 7×7). All ConvNet layers were designed using the same principles [42].
- **Xception:** Xception is a deep CNN architecture. It was developed by Google researchers in 2017 and inspired by Inception. Xception is founded on an 'extreme' performance of the Inception model, where Inception architectures have been changed with depthwise separable convolutions layers and consists of 3 major sections: Entry Flow, Middle Flow, and Exit Flow [43].

Transfer learning TL is a new research direction in the field of deep learning and machine learning. It is the amelioration of learning in a new task from the "target" dataset, through the transfer of knowledge from a related task that has already been learned from a "source" [44]. The learning process of TL is displayed in **Figure 1**.

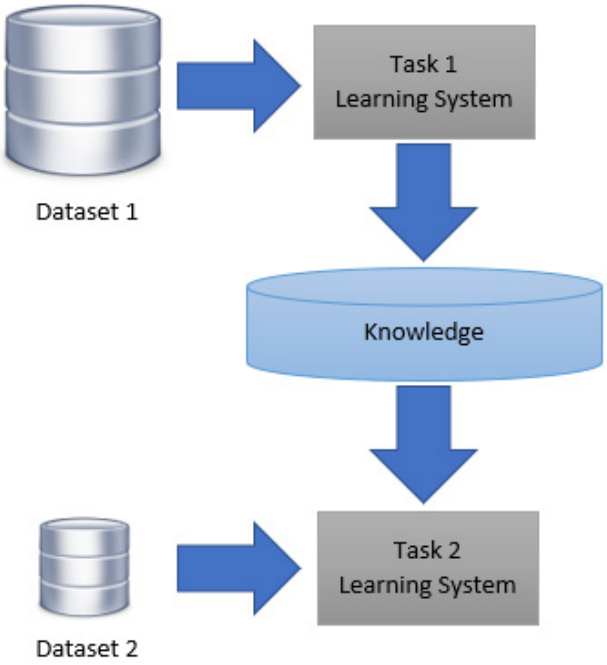

Figure 1 Learning process of TL.

TL received advanced attention from researchers and has been successfully applied to many fields like text classification and clustering, image classification and clustering, reinforcement learning, sentiment classification, and collaborative filtering [45]. Deep TL can be divided into four categories: Mapping-based, adversarial-based, instance-based, and network-based [46]. TL permits reusing knowledge from 1 problem domain in an allied domain. With TL, we can use a simple and powerful method called Featurizer for computing features using pre-trained DL models that transfers knowledge about favorable features from the original domain. We can perform featurization with deep TL combined with LR included in Apache Spark, based on TensorFlow and Keras [47].

TensorFlow and Keras

TensorFlow is an open-source machine learning library developed by Google and a tool for solving complex mathematical problems [48]. TensorFlow is a programming system in which the calculations are represented in a data flow graph. It can run faster than a pure Python program, supporting parallel computing, CPU, and GPU [49]. The TensorFlow-related code consists of 2 main phases, construction and execution. During the 1st stage of construction, the variables and the operations of the graph are determined and grouped. TensorFlow automatically manages the creation of the graph to allow optimization and parallelization of code and execution. The 2nd execution step uses a session to execute the operations of the graph. A graph will only perform operations after a session has been established. A session authorizes the placement of the operations of the graph in CPUs, GPUs, or TPUs and provides methods to execute them [50]. Keras is an open-source Python library that encapsulates access to functions offered by several machine learning libraries, including TensorFlow, Theano, or CNTK [51]. It is designed to create rapid models with deep neural networks. Keras use the Model and the Sequential APIs to build simple or very complex neural networks. Keras is recommended for rapid and simple prototyping. It supports recurrent networks and convolutional networks that, when combined, work transparently on CPU and GPU [52].

Material and methods

CT image Dataset

In this work, CT images were obtained from the open-source GitHub repository used for the diagnosis of COVID-19 [53]. The COVID-CT-Dataset has 349 CT images from people with COVID-19 and 397 from persons with non-COVID-19. The database was developed by [54] using images from various COVID19-related papers such as JAMA, NEJM, bioRxiv, Lancet, and medRxiv. **Figures 2** and **3** are chest CT images of COVID-19 and non-COVID-19, respectively.

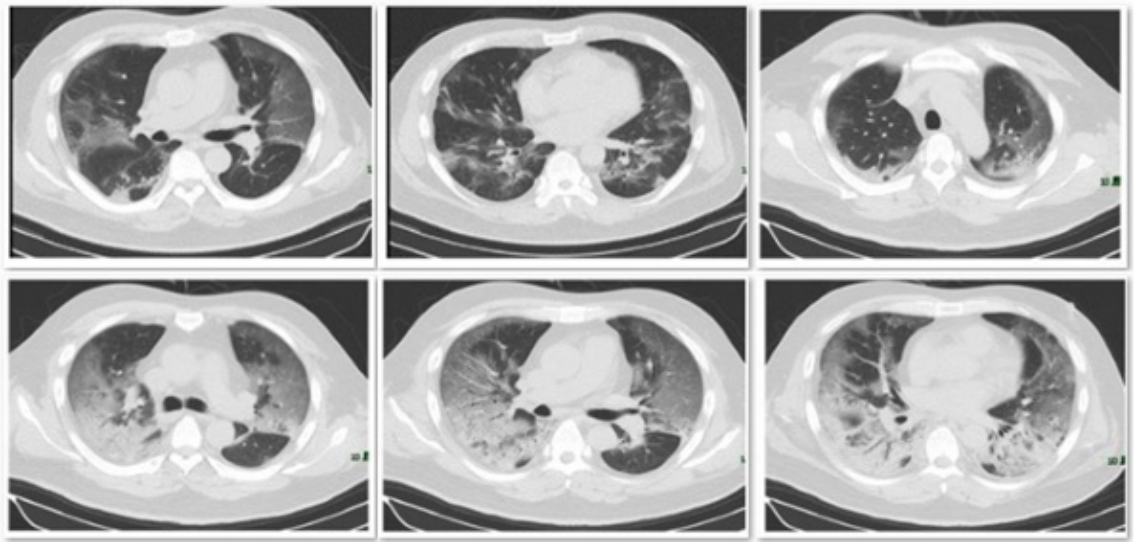

Figure 2 COVID-19 chest CT images.

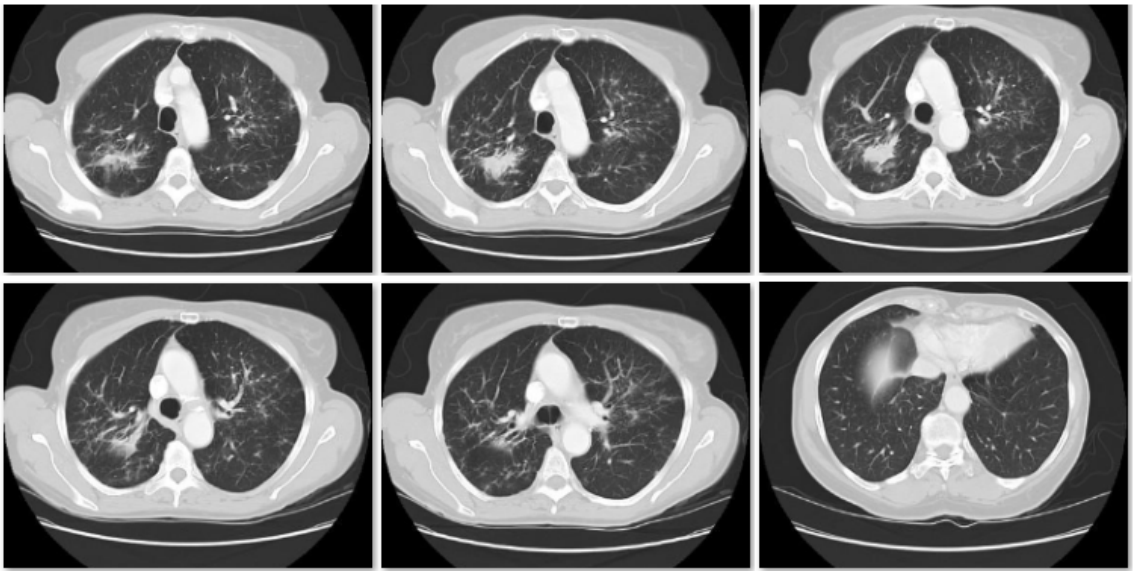

Figure 3 Non-COVID-19 chest CT images.

Apparatus

In this study, we used Deep Learning Pipelines that is a library published by Databricks Workspace (The Apache Spark-based analytics platform) to furnish high-level APIs for scalable deep learning and transfer learning via integration of popular deep learning libraries with MLlib Pipelines and Spark SQL. In this platform, we created and set-up a Cluster as:

- **Cluster Name:** COVID-CT-master;
- **Databricks Runtime Version:** Runtime 6.4 (Scala 2.11, Spark 2.4.5);
- **Instance:** 1 Driver: 15.3 GB Memory, 2 Cores, 1 DBU;
- **Availability Zone:** us-west-2c;
- **Spark Environment Variables:** PYSPARK version supports only Python 3;

We created 2 paths in Databricks File System (DBFS), the 1st one is DBFS/mL/COVID-CT-master/CTCOVID and the 2nd one is DBFS/mL/COVID-CT-master/CTNonCOVID to store respectively, the chest CT images affected by COVID-19 and the NonCOVID-19 images.

In this study, we trained and tested 3 different types of CNNs architectures with Apache Spark in the Databricks environment such as Xception, VGG16, and VGG19. We tried to detect–SARS-CoV-2 pneumonia in the chest CT images. For this reason, we used deep transfer learning Pipelines for Apache Spark and a combination with logistic regression. The general architecture of our work is summarized in **Figure 4**.

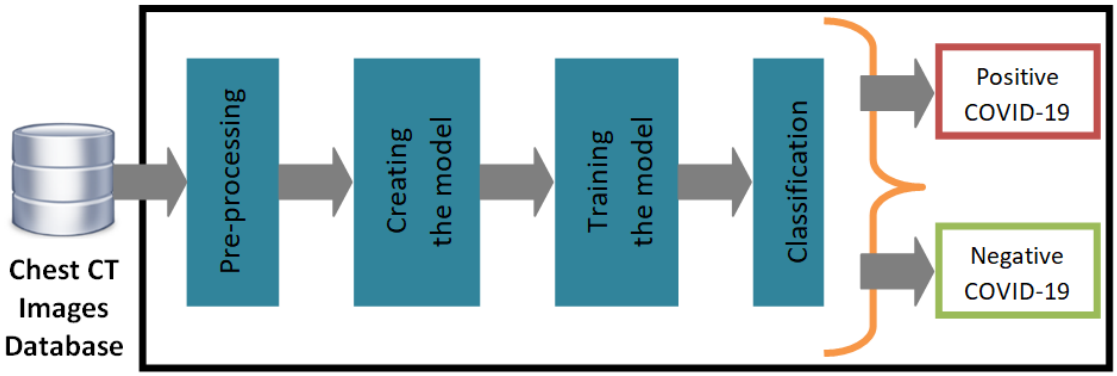

Figure 4 The general architecture of our model.

For our experience, the 1st stage to practice deep transfer learning on CT images is to load the images into a Spark DataFrame. This step has been done based on utility functions furnished by Spark and Deep Learning Pipelines that can load images and decode them in a distributed manner, allowing handling at scale. We attributed the values "1" and "0", respectively as labels for CT images affected by COVID-19 and NonCOVID-19. **Figures 5** and **6** show this transaction.

Figure 5 CT COVID-19 images with label 1.

Figure 6 CT NonCOVID-19 images with label 0.

To implement Transfer Learning on images, Deep Learning Pipelines furnish clear and simple utilities. This is the fastest method to code, run time, and employ deep learning on CT images. First, we have to split the dataset into two independent datasets for transfer learning. We used 70 and 30 % for training and testing DataFrames, respectively. Deep Learning Pipelines permit rapid transfer learning on Apache Spark cluster with the concept of a Featurizer. We have worked on the VGG16, VGG19, and Xception architectures. We have implemented a DeepImageFeaturizer. It has represented an essential

operation of Deep Learning Pipelines that automatically peels off the last layer of a CNN pre-trained model and inserts the three models' output as features for the logistic regression algorithm. **Figures 7 - 9** show the extraction of the program, which combines Logistic Regression, Pipeline, and DeepImageFeaturizer for the VGG16, VGG19, and Xception.

```
from pyspark.ml.classification import LogisticRegression
from pyspark.ml import Pipeline
from sparkdl import DeepImageFeaturizer

featurizer = DeepImageFeaturizer(inputCol="image", outputCol="features", modelName="VGG16")
lr = LogisticRegression(maxIter=20, regParam=0.05, elasticNetParam=0.3, labelCol="label")
p = Pipeline(stages=[featurizer, lr])

p_model = p.fit(trainDF)
```

```
▸ (38) Spark Jobs
Command took 16.12 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 5:49:31 PM on COVID-CT-master
```

Figure 7 Extraction of the program for VGG16.

```
from pyspark.ml.classification import LogisticRegression
from pyspark.ml import Pipeline
from sparkdl import DeepImageFeaturizer

featurizer = DeepImageFeaturizer(inputCol="image", outputCol="features", modelName="VGG19")
lr = LogisticRegression(maxIter=20, regParam=0.05, elasticNetParam=0.3, labelCol="label")
p = Pipeline(stages=[featurizer, lr])

p_model = p.fit(trainDF)
```

```
▸ (25) Spark Jobs
Command took 20.31 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 10:54:10 PM on COVID-CT-master
```

Figure 8 Extraction of the program for VGG19.

```
from pyspark.ml.classification import LogisticRegression
from pyspark.ml import Pipeline
from sparkdl import DeepImageFeaturizer

featurizer = DeepImageFeaturizer(inputCol="image", outputCol="features", modelName="Xception")
lr = LogisticRegression(maxIter=20, regParam=0.05, elasticNetParam=0.3, labelCol="label")
p = Pipeline(stages=[featurizer, lr])

p_model = p.fit(trainDF)
```

```
▸ (31) Spark Jobs
Command took 12.41 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 9:48:38 PM on COVID-CT-master
```

Figure 9 Extraction of the program for Xception.

In this phase, we check the performance of our deep learning model. There are 4 categories of results that could occur when making classification predictions. These terms are True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). Note that:

- TP: When we predict a positive observation and it is in reality positive.
- FP: When we predict a positive observation and it is in reality negative.
- TN: When we predict a negative observation and it is in reality negative.
- FN: When we predict a negative observation and it is in reality positive.

To evaluate our model, we have used 4 criteria performance measures: Precision, Recall, F1-Score, and Accuracy.

- Precision is defined as the percentage of positive instances out of the total predicted positive instances. It is measured as:

$$\frac{TP}{TP + FP}$$

- Recall (Sensitivity or True Positive Rate) is calculated as the percentage of positive instances out of the total actual positive instances. It is defined as:

$$\frac{TP}{TP + FN}$$

- F1-Score is defined as the harmonic mean of precision and recall. It is measured as:

$$\frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * Precision * Recall}{Precision + Recall}$$

- Accuracy is referred to as the percentage of correct predictions for the test data. It can be calculated as:

$$\frac{TP + TN}{TP + FP + TN + FN}$$

Results and discussion

We tested the first 3 indices; Precision, Recall, and F1-Score to evaluate the performance of our model using the 3 pre-trained CNNs models VGG16, VGG19, and Xception. **Figure 10** shows the results of this experiment.

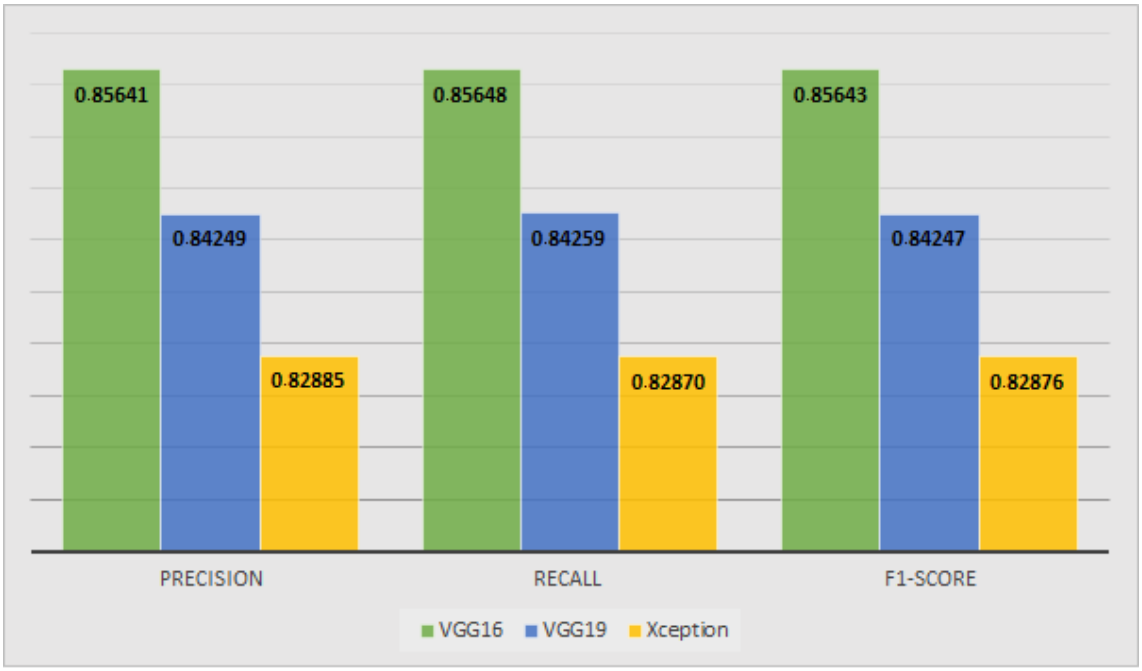

Figure 10 Precision, Recall, and F1-score for VGG16, VGG19, and Xception.

According to **Figure 10**, we can easily remark that VGG16 achieves the best results 85.6413, 85.648, and 85.6428 % for Precision, Recall, and F1-Score, respectively. For the same order concerning the performance index, we find in 2nd place VGG19 with 84.2493, 84.2592 and 84.2470 %. Finally, in 3rd place, we have Xception with 82.8849, 82.8703 and 82.8759 %.

The last index tested is the accuracy. With VGG16, our model achieved an accuracy of 85.6481 % which is recorded in **Figure 11**. On the other hand, with VGG16, our architecture achieves an accuracy of 84.2592 % which is displayed in **Figure 12**. Finally, with Xception, our model reached the accuracy of 82.8703 % which is illustrated in **Figure 13**.

```
1 from pyspark.ml.evaluation import MulticlassClassificationEvaluator
2
3 tested_df = p_model.transform(testDF)
4 evaluator = MulticlassClassificationEvaluator(metricName="accuracy")
5 print("Accuracy = " + str(evaluator.evaluate(tested_df.select("prediction", "label"))))
```

```
▸ (2) Spark Jobs
▸ tested_df: pyspark.sql.dataframe.DataFrame = [image: struct, label: integer ... 4 more fields]
Accuracy = 0.8564814814814815
Command took 14.17 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 5:49:31 PM on COVID-CT-master
```

Figure 11 Training accuracy for VGG16.

```
1 from pyspark.ml.evaluation import MulticlassClassificationEvaluator
2
3 tested_df = p_model.transform(testDF)
4 evaluator = MulticlassClassificationEvaluator(metricName="accuracy")
5 print("Accuracy = " + str(evaluator.evaluate(tested_df.select("prediction", "label"))))
```

```
▸ (2) Spark Jobs
▸ tested_df: pyspark.sql.dataframe.DataFrame = [image: struct, label: integer ... 4 more fields]
Accuracy = 0.8425925925925926
Command took 17.59 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 10:54:10 PM on COVID-CT-master
```

Figure 12 Training accuracy for VGG19.

```
1 from pyspark.ml.evaluation import MulticlassClassificationEvaluator
2
3 tested_df = p_model.transform(testDF)
4 evaluator = MulticlassClassificationEvaluator(metricName="accuracy")
5 print("Accuracy = " + str(evaluator.evaluate(tested_df.select("prediction", "label"))))
```

```
▸ (2) Spark Jobs
▸ tested_df: pyspark.sql.dataframe.DataFrame = [image: struct, label: integer ... 4 more fields]
Accuracy = 0.8287037037037037
Command took 10.48 minutes -- by houssam.benbrahim@uit.ac.ma at 9/5/2020, 9:48:38 PM on COVID-CT-master
```

Figure 13 Training accuracy for Xception.

In this paper, a method was created on deep transfer learning using CNN-based VGG16, VGG19, and Xception with Apache Spark, Keras TensorFlow, and Logistic Regression for the detection of COVID-19 in chest CT images. The experimental results of our model reached an accuracy of 85.6481, 84.2592, and 82.8703 % for VGG16, VGG19, and Xception, respectively. We also tested other performance measurements as F1-Score, Precision, and Recall. All the latest indices have achieved high values ranging from 82 % up to 86 %.

In a similar study [11], the authors have created a DL algorithm based on modified Inception (M-Inception) using CT images to screen for Corona Virus Disease. The external testing showed a total accuracy of 73 % with a specificity of 67 % and a sensitivity of 74%. In another work [55], the authors proposed A Light CNN design based on the model of the SqueezeNet for detecting COVID-19 from CT scans of the chest. The proposed modified SqueezeNet CNN attends 83.00 % accuracy, 85.00 % of sensitivity, 81.00 % of specificity, 81.73 % of precision, and 83.33 % of F1Score. In [56], the authors created a transfer learning technique for the classification of COVID-19. They used DenseNet-121 to train the deep learning network by removing the gradient problem. The accuracy obtained from this technique was 87 %. In another paper [57], the authors tested the pre-trained DL method to detect COVID-19 infection in lung images. This method is implemented over 1,266 patients from 6 different cities. The accuracy achieved from this study was 87 %.

In this research, we can easily notice that deep transfer learning Pipelines with Apache Spark using advanced methods as pre-trained CNN VGG16, VGG19, and Xception models, combined with the logistic regression algorithm achieved important results for detecting COVID-19 in chest CT images. This exceeds an accuracy of 85 %, which is similar or better than several other studies.

Conclusions

This study has presented deep transfer learning Pipelines based on 3 CNN pre-trained models, VGG16, VGG19, and Xception, to detect and classify COVID-19 cases from CT images automatically. Due to the availability of the CT image dataset in the open-source GitHub repository, we have exploited this opportunity to elaborate our experience in the Databricks workspace. In this study, we stocked the database and used the advanced performance of Apache Spark and Keras TensorFlow coupled with the logistic regression algorithm. Our developed system can perform an accuracy of 85.6481, 84.2592, and 82.8703 % for VGG16, VGG19, and Xception, respectively. We can claim that our architecture can provide a promising solution to detect Covid-19 disease in chest CT images with high performance. Our future work aims to create a new platform for triple detection of the novel coronavirus 2019 using Machine learning and deep learning combined with Apache Spark, the big data analysis framework. This classification will be presented in 3 parts: Using pre-defined standard symptoms, X-ray images, and CT images.

Acknowledgements

The authors are grateful to the reviewers for the evaluation of the manuscript.

References

[1] JFW Chan, S Yuan, KH Kok, KKW To, H Chu, J Yang, F Xing, J Liu, CCY Yip, RWS Poon, HW Tsoi, SKF Lo, KH Chan, VKM Poon, WM Chan, JD Ip, JP Cai, VCC Cheng, H Chen, CKM Hui and KY Yuen. A familial cluster of pneumonia associated with the 2019 novel coronavirus indicating person-to-person transmission: A study of a family cluster. *Lancet* 2020; **395**, 514-23.

[2] N Zhu, D Zhang, W Wang, X Li, B Yang, J Song, X Zhao, B Huang, W Shi, R Lu, P Niu, F Zhan, X Ma, D Wang, W Xu, G Wu, GF Gao, D Phil and W Tan. A novel coronavirus from patients with pneumonia in China, 2019. *N. Engl. J. Med.* 2020; **382**, 727-33.

[3] World Health Organization, Coronavirus disease (COVID-19), Available at: https://covid19.who.int, accessed September 2020.

[4] Ministry of Health of Morocco. The official portal of Coronavirus in Morocco, Available at: http://www.covidmaroc.ma/Pages/LESINFOAR.aspx, accessed September 2020.

[5] W Wang, J Tang and F Wei. Updated understanding of the outbreak of 2019 novel coronavirus (2019-nCoV) in Wuhan, China. *J. Med. Virol.* 2020; **92**, 441-7.

[6] M Shen, Y Zhou, J Ye, AAA Al-Maskri, Y Kang, S Zeng and S Cai. Recent advances and perspectives of nucleic acid detection for coronavirus. *J. Pharm. Anal.* 2020; **10**, 97-101.

[7] T Ai, Z Yang, H Hou, C Zhan, C Chen, W Lv, Q Tao, Z Sun and L Xia. Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: A report of 1014 cases. *Radiology* 2020; **296**, E32-E40.

[8] Y Fang, H Zhang, J Xie, M Lin, L Ying, P Pang and W Ji. Sensitivity of chest CT for COVID-19: Comparison to RT-PCR. *Radiology* 2020; **296**, E115-E117.

[9] JP Kanne, BP Little, JH Chung, BM Elicker and LH Ketai. Essentials for radiologists on COVID-19: An update-radiology scientific expert panel. *Radiology* 2020; **296**, E113-E114.

[10] X Xie, Z Zhong, W Zhao, C Zheng, F Wang and J Liu. Chest CT for typical 2019-nCoV pneumonia: Relationship to negative RT-PCR testing. *Radiology* 2020; **296**, E41-E45.

[11] S Wang, B Kang, J Ma, X Zeng, M Xiao, J Guo, M Cai, J Yang, Y Li, X Meng and B Xu. A deep learning algorithm using CT images to screen for Corona Virus Disease (COVID-19). *Eur. Radiol.* 2021. https://doi.org/10.1007/s00330-021-07715-1.

[12] O Gozes, M Frid-Adar, H Greenspan, PD Browning, H Zhang, W Ji, A Bernheim and E Siegel. Rapid AI development cycle for the coronavirus (covid-19) pandemic: Initial results for automated detection & patient monitoring using deep learning CT image analysis. arXiv preprint 2020. Available at: https://arxiv.org/abs/2003.05037, accessed September 2020.

[13] A Abbas, MM Abdelsamea and MM Gaber. Classification of COVID-19 in chest X-ray images using DeTraC deep convolutional neural network. *Appl. Intell.* 2021; **51**, 854-64.
[14] L Wang and A Wong. COVID-Net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images. *Sci. Rep.* 2020; **10**, 1-12.
[15] H Benbrahim, H Hachimi and A Amine. Deep transfer learning with Apache Spark to detect COVID-19 in chest X-ray images. *Rom. J. Inf. Sci. Tech.* 2020; **23**, S117-S129.
[16] AA Ardakani, AR Kanafi, UR Acharya, N Khadem and A Mohammadi. Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks. *Comput. Biol. Med.* 2020; **121**, 103795.
[17] CM Bishop. *Pattern recognition and machine learning*. Springer, New York, 2006, p. 738.
[18] Y Anzai. *Pattern recognition and machine learning*. Elsevier, Amsterdam, 2012, p. 407.
[19] H Wang, C Ma and L Zhou. A brief review of machine learning and its application. *In:* Proceedings of the International Conference on Information Engineering and Computer Science, Wuhan, China. IEEE. 2009, p. 1-4.
[20] J Watt, R Borhani and A Katsaggelos. *Machine learning refined: Foundations, algorithms, and applications*, Cambridge University Press, Cambridge, 2016, p. 286.
[21] E Alpaydin. *Introduction to machine learning.* MIT press, Cambridge, 2020, p. 712.
[22] CYJ Peng, KL Lee and GM Ingersoll. An introduction to logistic regression analysis and reporting. *J. Educ. Res.* 2002; **96**, 3-14.
[23] JI Hoffman. *Biostatistics for medical and biomedical practitioners.* Academic Press, California, 2015, p. 734.
[24] FE Harrell. *Binary logistic regression. In:* Regression modeling strategies. Springer, Cham. 2015, p. 219-74.
[25] A Tattersall and MJ Grant. Big data: What is it and why it matters. *Health Info. Libr. J.* 2016; **33**, 89-91.
[26] LV Satyanarayana. A Survey on challenges and advantages in big data. *Int. J. Comput. Sci. Inf. Technol.* 2015; **6**, 115-9.
[27] K Wangand MMH Khan. Performance prediction for apache spark platform. *In:* Proceedings of the 17th International Conference on High Performance Computing and Communications, 7th International Symposium on Cyberspace Safety and Security, and 12th International Conference on Embedded Software and Systems. IEEE, 2015, p. 166-73.
[28] M Frampton. *Mastering apache spark*. Packt Publishing, Birmingham, 2015, p. 318.
[29] HP Sahana, MS Sanjana, NM Muddasir and KP Vidyashree. Apache spark methods and techniques in big data: A review. *In:* Proceedings of the Inventive Communication and Computational Technologies. Springer, Singapore. 2020, p. 721-6.
[30] I Goodfellow, Y Bengio, A Courville and Y Bengio. *Deep learning.* Vol I. MIT press, Cambridge, 2016, p. 800.
[31] NF Hordri, SS Yuhaniz and SM Shamsuddin. Deep learning and its applications: A review. *In:* Proceedings of Conference on Postgraduate Annual Research on Informatics Seminar, Kuala Lumpur, 2016.
[32] Y Bengio, A Courville and P Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* 2013; **35**, 1798-828.
[33] JG Lee, S Jun, YW Cho, H Lee, GB Kim, JB Seo and N Kim. Deep learning in medical imaging: General overview. *Korean J. Radiol.* 2017; **18**, 570-84.
[34] S Rao. Building a data pipeline for deep learning. NetApp 2019, Available at: lenovonetapp.com/pdf/wp-7299.pdf, accessed September 2020.
[36] H El-Amir and M Hamdy. *Deep learning pipeline: Building a deep learning model with TensorFlow*. Apress, New York, 2019, p. 576.
[36] Databricks. A vision for making deep learning simple, Available at: https://databricks.com/blog/2017/06/06/databricks-vision-simplify-large-scale-deep-learning.html, accessed September 2020.
[37] Y Kim. Convolutional neural networks for sentence classification. *In*: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Doha, Qatar, 2014, p. 1746-51.

[38] A Bhandare, M Bhide, P Gokhale and R Chandavarkar. Applications of convolutional neural networks. *Int. J. Comput. Sci. Inf. Technol.* 2016; **7**, 2206-15.
[39] A Hidaka and T Kurita. Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks. *In*: Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications, Kusatsu, Japan, 2017, p. 160-7.
[40] A Khan, A Sohail, U Zahoora and AS Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artif. Intell. Rev.* 2020; **53**, 1-62.
[41] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. *In:* Proceedings of the International Conference on Learning Representations, California, USA, 2015.
[42] M Jaderberg, K Simonyan, K Kavukcuoglu and A Zisserman. *Spatial transformer networks*. *In:* Advances in neural information processing systems. Google DeepMind, London, 2015, p. 2017-25.
[43] F Chollet. Xception: Deep learning with depthwise separable convolutions. *In:* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Hawaii, USA, 2017, p. 1251-8.
[44] F Zhuang, Z Qi, K Duan, D Xi, Y Zhu, H Zhu, H Xiong and Q He. A comprehensive survey on transfer learning. *Proc. IEEE* 2020; **109**, 43-76.
[45] Z Fuzhen and L Ping. Some applications in transfer learning, Available at: http://www.intsci.ac.cn/users/zhuangfuzhen/TransferApplication.pdf, accessed September 2020.
[46] C Tan, F Sun, T Kong, W Zhang, C Yang and C Liu. A survey on deep transfer learning. *In*: Proceedings of the International Conference on Artificial Neural Networks. Springer, Cham. 2018, p. 270-9.
[47] Databricks. Featurization for transfer learning, Available at: https://docs.databricks.com/ applications/machine-learning/preprocess-data/transfer-learning-tensorflow.html, accessed September 2020.
[48] M Abadi, A Agarwal, P Barham, E Brevdo, Z Chen, C Citro, GS Corrado, A Davis, J Dean, M Devin, S Ghemawat, I Goodfellow, A Harp, G Irving, M Isard, Y Jia, R Jozefowicz, L Kaiser, M Kudlur, J Levenberg, D Mane, R Monga, S Moore, D Murray, C Olah, M Schuster, J Shlens, B Steiner, I Sutskever, K Talwar, P Tucker, V Vanhoucke, V Vasudevan, F Viegas, O Vinyals, P Warden, M Wattenberg, M Wicke, Y Yu and X Zheng. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint 2016. Available at: https://arxiv.org/abs/1603.04467, accessed September 2020.
[49] MT Martinez. *An overview of Google's machine intelligence software TensorFlow*. Sandia National, New Mexico, 2016.
[50] G Zaccone. *Getting started with TensorFlow*. Packt Publishing, Birmingham, 2016, p. 180.
[51] NK Manaswi. *Understanding and working with Keras. In:* Deep learning with applications using python. Springer, California, 2018, p. 31-43.
[52] A Gulli and S Pal. *Deep learning with Keras.* Packt Publishing, Birmingham, 2017, p. 318.
[53] J Zhao, Y Zhang, X He and P Xie. COVID-CT-Dataset: A CT scan dataset about COVID-19. arXiv preprint 2020. Available at: https://arxiv.org/abs/2003.13865, accessed September 2020.
[54] Github. COVID CT, Available at: https://github.com/UCSD-AI4H/COVID-CT, accessed September 2020.
[55] M Polsinelli, L Cinque and G Placidi. A light CNN for detecting COVID-19 from CT scans of the chest. *Pattern Recognit. Lett.* 2020; **140**, 95-100.
[56] L Sarker, MM Islam, T Hannan and Z Ahmed. COVID-DenseNet: A deep learning architecture to detect COVID-19 from chest radiology images. *Preprint* 2020. https://doi.org/10.20944/preprints202005.0151.v1
[57] S Wang, YZha, W Li, Q Wu, X Li, M Niu, M Wang, X Qiu, H Li, H Yu, and W Gong. A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis. *Eur. Respir. J.* 2020; **56**, 2000775.